Manuscript Title

This manuscript (permalink) was automatically generated from greenelab/generalization-manuscript@b82f5e9 on July 20, 2023.

Authors

- **John Doe**
  XXXX-XXXX-XXXX-XXXX · johndoe · johndoe · @johndoe@mastodon.social
  Department of Something, University of Whatever · Funded by Grant XXXXXXXX

- **Jane Roe** ✉
  XXXX-XXXX-XXXX-XXXX · janeroe
  Department of Something, University of Whatever; Department of Whatever, University of Something

✉ — Correspondence possible via GitHub Issues or email to Jane Roe <jane.roe@whatever.edu>.

Abstract

Methods

Mutation data download and preprocessing

To generate binary mutated/non-mutated gene labels for our machine learning model, we used mutation calls for TCGA samples from MC3 [1] and copy number threshold calls from GISTIC2.0 [2]. MC3 mutation calls were downloaded from the Genomic Data Commons (GDC) of the National Cancer Institute, at https://gdc.cancer.gov/about-data/publications/pancanatlas. Thresholded copy number calls are from an older version of the GDC data and are available here: https://figshare.com/articles/dataset/TCGA_PanCanAtlas_Copy_Number_Data/6144122. We removed hypermutated samples, defined as two or more standard deviations above the mean non-silent somatic mutation count, from our dataset to reduce the number of false positives (i.e., non-driver mutations). Any sample with either a non-silent somatic variant or a copy number variation (copy number gain in the target gene for oncogenes and copy number loss in the target gene for tumor suppressor genes) was included in the positive set; all remaining samples were considered negative for mutation in the target gene.

We followed a similar procedure to generate binary labels for cell lines from CCLE, using the data available on the DepMap download portal at https://depmap.org/portal/download/all/. Mutation information was retrieved from the `OmicsSomaticMutations.csv` data file, and copy number inforamtion was retrieved from the `OmicsCNGene.csv` data file. We thresholded the CNV log-ratios provided by CCLE into binary gain/loss calls using a lower threshold of $\log_2(3/2)$ (i.e. cell lines with a log-ratio below this threshold were considered to have a full copy loss in the corresponding gene), and an upper threshold of $\log_2(5/2)$ (i.e. cell lines with a log-ratio above this threshold were considered to have a full copy gain in the corresponding gene). After applying the same hypermutation criteria that we used for TCGA, no cell lines in CCLE were identified as hypermutated. After preprocesing, 1402 cell lines with mutation and copy number data remained. We then combined non-silent point mutations and copy number gain/loss information into binary labels using the same criteria as for TCGA.

Gene expression data download and preprocessing

RNA sequencing data for TCGA was downloaded from GDC at the same link provided above for the Pan-Cancer Atlas. We discarded non-protein-coding genes and genes that failed to map, and removed tumors that were measured from multiple sites. After filtering to remove hypermutated samples and taking the intersection of samples with both mutation and gene expression data, 9074 TCGA samples remained.

RNA sequencing data for CCLE was downloaded from the DepMap download portal, linked above, in the `CCLE_expression.csv` data file. After taking the intersection of CCLE cell lines with both mutation and gene expression data, 1402 cell lines remained. For experiments making predictions across datasets (i.e., training models on TCGA and evaluating performance on CCLE, or vice-versa) we took the intersection of genes in both datasets, resulting in 16041 gene features. For experiments where only TCGA data was used (i.e., evaluating models on held-out cancer types), we used all 16148 gene features present in TCGA after the filtering described above.

Cancer gene set construction

In order to study mutation status classification for a diverse set of cancer driver genes, we started with the set of 125 frequently altered genes from Vogelstein et al. [3] (all genes from Table S2A). For each target gene, to ensure that the training dataset was reasonably balanced (i.e., that there would be enough mutated samples to train an effective classifier), we included only cancer types with at least 15 mutated samples and at least 5% mutated samples, which we refer to here as “valid” cancer types. In some cases, this resulted in genes with no valid cancer types, which we dropped from the analysis. Out of the 125 genes originally listed in the Vogelstein et al. cancer gene set, we retained 71 target genes for the TCGA to CCLE analysis, and 70 genes for the CCLE to TCGA analyses. For these analyses, each gene needed at least one valid cancer type in TCGA and one valid cancer type in CCLE, to construct the train and test sets. For the cancer type holdout analysis, we retained 56 target genes: in this case, each gene needed at least two valid cancer types in TCGA to be retained, one to train on and one to hold out.

Classifier setup and cross-validation design

We trained logistic regression classifiers to predict whether or not a given sample had a mutational event in a given target gene using gene expression features as explanatory variables. Our model was trained on gene expression data (X) to predict somatic mutation presence or absence (y) in a target gene. To control for varying mutation burden per sample and to adjust for potential cancer type-specific expression patterns, we included one-hot encoded cancer type and $\log_{10}$(sample mutation count) in the model as covariates. Since gene expression datasets tend to have many dimensions and comparatively few samples, we used a LASSO penalty to perform feature selection [4]. LASSO logistic regression has the advantage of generating sparse models (some or most coefficients are 0), as well as having a single tunable hyperparameter which can be easily interpreted as an indicator of regularization strength/model simplicity.

LASSO (ℓ_1-penalized) logistic regression finds the feature weights $\hat{w} \in \mathbb{R}^p$ solving the following optimization problem:

$$\hat{w} = \mathrm{argmin}_w \; (C \cdot l(X, y; w)) + ||w||_1$$

where $i \in \{1, \dots, n\}$ denotes a sample in the dataset, $X_i \in \mathbb{R}^p$ denotes features (gene expression measurements) from the given sample, $y_i \in \{0, 1\}$ denotes the label (mutation presence/absence) for the given sample, and $l(\cdot)$ denotes the negative log-likelihood of the observed data given a particular choice of feature weights, i.e.

$$l(X, y; w) = -\sum_{i=1}^{n} y_i \log\left(\frac{1}{1 + e^{-w^\top X_i}}\right) + (1 - y_i) \log\left(1 - \frac{1}{1 + e^{-w^\top X_i}}\right)$$

Given weight values $\hat{w}$, it is straightforward to predict the probability of a positive label (mutation in the target gene) $P(y^* = 1 \mid X^*; \hat{w})$ for a test sample X^*:

$$P(y^* = 1 \mid X^*; \hat{w}) = \frac{1}{1 + e^{-\hat{w}^\top X^*}}$$

and the probability of no mutation in the target gene, $P(y^* = 0 \mid X^*; \hat{w})$, is given by (1 - the above quantity).

This optimization problem leaves one hyperparameter to select: C, which controls the inverse of the strength of the L1 penalty on the weight values (i.e. regularization strength scales with $\frac{1}{C}$). Although the LASSO optimization problem does not have a closed form solution, the loss function is convex, and iterative optimization algorithms are commonly used for finding reasonable solutions. For fixed values of C, we solved for $\hat{w}$ using `scikit-learn`'s `LogisticRegression` method [5], which uses the coordinate descent optimization method implemented in `liblinear` [6]. We selected this implementation rather than the `SGDClassifier` stochastic gradient descent implementation because coordinate descent/ `liblinear` tends to generate sparser models and does not depend on a learning rate parameter, although after hyperparameter tuning performance is generally comparable between the implementations [7].

To assess model selection across contexts (datasets and cancer types), we trained models using a variety of LASSO parameters on 75% of the training dataset, holding out 25% of the training dataset as the "cross-validation" set and also evaluating across contexts as the "test" set. We trained models using C values evenly spaced on a logarithmic scale between (10^{-3}, 10^{7}); i.e. the output of `numpy.logspace(-3, 7, 21)`. This range was intended to give evenly distributed coverage across genes and cancer types that included "underfit" models (predicting only the mean or using very few features, poor performance on all datasets), "overfit" models (performing perfectly on training data but comparatively poorly on cross-validation and test data), and a wide variety of models in between that typically included the best fits to the cross-validation and test data. To assess variability between train/CV splits, we used all 4 splits (25% holdout sets) x 2 random seeds for a total of 8 different training sets for each gene, using the same test set (i.e. all of the held-out context, either one cancer type or one dataset) in each case.

"Best model" vs. "smallest good model" analysis details

Description of "smallest good" heuristic Statistical testing?

Neural network setup and parameter selection

Inspired by the intermediate-complexity model in [8], as a tradeoff between computational cost and ability to represent non-linear decision boundaries, we trained a three-layer fully connected neural network with ReLU nonlinearities [9] to predict mutation status. For the experiments described in the main paper, we varied the size of the first hidden layer in the range {1, 2, 3, 4, 5, 10, 50, 100, 500, 1000}. We fixed the size of the second hidden layer to be half of the size of the first hidden layer, rounded up to the nearest integer, and the size of the third hidden layer was the number of classes, 2 in our case. Our models were trained for 100 epochs of mini-batch stochastic gradient descent in PyTorch [10], using the Adam optimizer [11] and a fixed batch size of 50. To select the remaining hyperparameters for each hidden layer size, we performed a random search over 10 combinations, with a single train/test split stratified by cancer type, using the following hyperparameter ranges: learning rate {0.1, 0.01, 0.001, 5e-4, 1e-4}, dropout proportion {0.1, 0.5, 0.75}, weight decay (L2 penalty) {0, 0.1, 1, 10, 100}. We used the same train/cross-validation split strategy described above, generating 8 different performance measurements for each gene and hidden layer size, for the neural network experiments as well.

For the *EGFR* gene, we also ran experiments where we varied the dropout proportion and the weight decay hyperparameter as the regularization axis, and selected the remaining hyperparameters (including the hidden layer size) using a random search. In these cases, we used a fixed range for dropout of {0.0, 0.05, 0.125, 0.25, 0.375, 0.5, 0.625, 0.75, 0.875, 0.95}, and a fixed range for weight decay of {0.0, 0.001, 0.005, 0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.75, 1.0, 10.0}. All neural network analyses were performed on a Ubuntu 18.04 machine with a NVIDIA RTX 2060 GPU.

Open science/reproducibility stuff

Results

Evaluating model generalization using public cancer data

We collected data from the TCGA Pan-Cancer Atlas and the Cancer Cell Line Encyclopedia to predict the presence or absence of mutations in cancer genes, as a benchmark of cancer-related information content across cancer types and contexts. We trained mutation status classifiers across approximately 70 genes involved in cancer development and progression from Vogelstein et al. 2013 [12], using LASSO logistic regression with gene expression (RNA-seq) values as predictive features. We designed experiments to evaluate the generalization of mutation status classifiers across datasets (TCGA to CCLE and CCLE to TCGA) and across biological contexts (cancer types) within TCGA, relative to a within-dataset baseline (Figure 1).

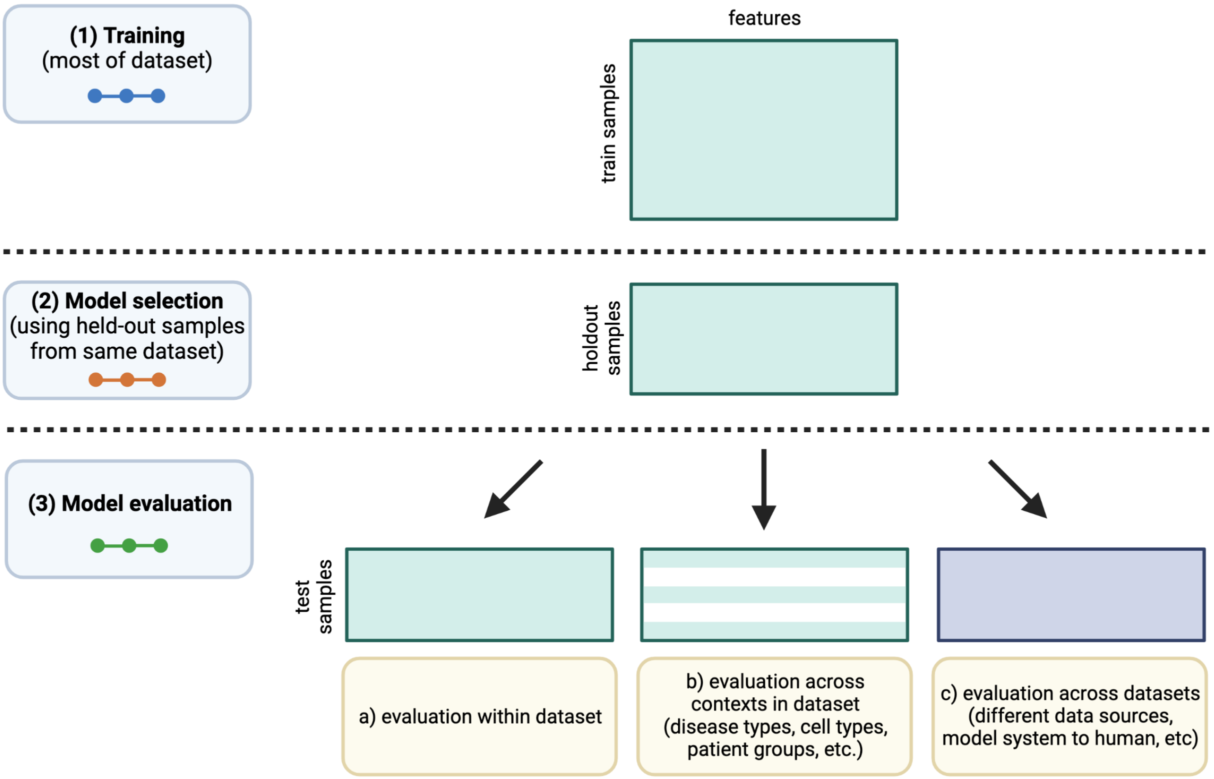
(1) Training
(most of dataset)
features
train samples
(2) Model selection
(using held-out samples
from same dataset)
holdout samples
(3) Model evaluation
test samples
a) evaluation within dataset
b) evaluation across contexts in dataset (disease types, cell types, patient groups, etc.)
c) evaluation across datasets (different data sources, model system to human, etc)

Figure 1: Schematic of experimental design. The colors of the "dots" in the training/model selection/model evaluation panels on the left correspond to train/CV/test curves in the following results figures.

Generalization from human tumor samples to cell lines is more effective than the reverse

To evaluate "cross-dataset" generalization, we trained mutation status classifiers on human tumor data from TCGA and evaluated them on cell line data from CCLE, as well as the reverse from CCLE to TCGA. As an example, we examined *EGFR*, an oncogenic tyrosine kinase that is commonly mutated in diverse cancer types and cancer cell lines, including lung cancer, colorectal cancer, and glioblastoma [13,14]. For EGFR mutation status classifiers trained on TCGA and evaluated on CCLE, we saw that AUPR on cell lines was slightly worse than on held-out tumor samples, but comparable across regularization levels/LASSO parameters (Figure 2A). On the other hand, EGFR classifiers trained on CCLE and evaluated on TCGA performed considerably worse on human tumor samples as compared to held-out cell lines (Figure 2B).

To explore these tendencies more generally, we compared performance across all genes in the Vogelstein et al. dataset, for both TCGA to CCLE and CCLE to TCGA generalization. We measured the difference between performance on the holdout data within the training dataset and performance across datasets, with a positive difference indicating poor generalization (better holdout performance than test performance) and a 0 or negative difference indicating good generalization (comparable test performance to holdout performance). For generalization from TCGA to CCLE, we observed that median AUPR differences were mostly centered around 0 for most genes, with some exceptions at the extremes (Figure 2C; performance differences on the y-axis). An example of a gene exhibiting poor generalization was *IDH1*, the leftmost gene in Figure 2C, with good performance on held-out TCGA data and poor performance on CCLE data. IDH-mutant glioma cell lines are poorly represented compared to IDH-mutant patient tumors, which may explain the difficulty of generalization to cell lines for *IDH1* mutation classifiers [15]. For generalization from CCLE to TCGA, we observed a more pronounced upward shift toward better performance on CCLE and worse on TCGA, with most genes performing better on the CCLE holdout data and very few genes generalizing comparably to the TCGA samples (Figure 2D).

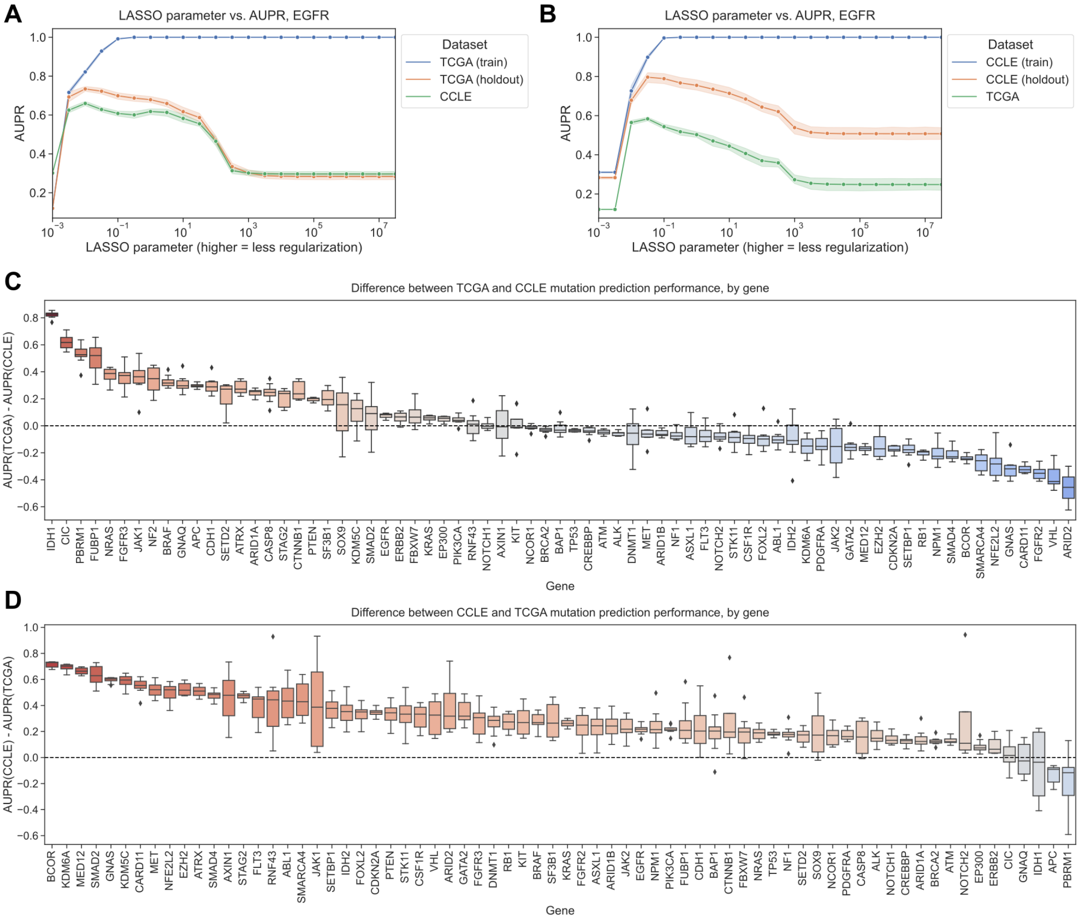

Figure 2: **A.** *EGFR* mutation status prediction performance on training samples from TCGA (blue), held-out TCGA samples (orange), and CCLE samples (green), across varying LASSO parameters. **B.** *EGFR* mutation status prediction performance on training samples from CCLE (blue), held-out CCLE samples (orange), and TCGA samples (green). **C.** Difference in mutation status prediction performance for models trained on TCGA (holdout data) and evaluated on CCLE (test data), across 71 genes from Vogelstein et al. For each gene, the best model (LASSO parameter) was selected using holdout AUPR performance. Genes on x-axis are ordered by median AUPR difference across cross-validation splits, from highest to lowest. **D.** Difference in mutation status prediction performance for models trained on CCLE (holdout data) and evaluated on TCGA (test data), across 70 genes from Vogelstein et al.

“Best” and “smallest good” model selection strategies perform comparably

To address the question of whether more parsimonious models tend to generalize better or not, we designed two model selection schemes and compared them for the TCGA to CCLE and CCLE to TCGA mutation prediction problems (Figure 3A). The "best" model selection scheme chooses the top-performing model/LASSO parameter on the holdout dataset from the same source as the training data and applies it to the test data from the other data source. The intention of the "smallest good" model selection scheme is to balance parsimony with reasonable performance on the holdout data, since simply selecting the smallest possible model (generally, the dummy regressor/mean predictor) is not likely to generalize well. To accomplish this, we first identify the top 25% of well-performing models on the holdout dataset; then, from this subset of models, we choose the smallest (i.e., highest LASSO parameter) to apply to the test data. In both cases, we exclusively use the holdout data to select a model and only apply the model to out-of-dataset samples to evaluate generalization performance *after* model selection.

For TCGA to CCLE generalization, 27/71 genes (38.0%) had better performance for the "best" model, and 17/71 genes (23.9%) had better generalization performance with the "smallest good" model. The other 27 genes had the same "best" and "smallest good" model (in other words, the "smallest good" model was also the best-performing overall, and the difference was 0) (Figure 3B). For CCLE to TCGA generalization, 23/70 genes (32.9%) had better performance for the "best" model and 18/70 (25.7%) for the "smallest good," with the other 29 having the same model fulfill both criteria (Figure 3C). Overall, these results do not support the hypothesis that the most parsimonious model generalizes the best: for both generalization problems there are slightly more genes where the best-performing model on the holdout dataset is also the best-performing on the test set, although there are some genes where the "smallest good" approach works well.

We examined genes that fell into either category for TCGA to CCLE generalization (dotted lines on Figure 3B). For *NF1*, the "best" model outperforms the "smallest good" model (Figure 3D). Comparing holdout (orange) and cross-dataset (green) performance, both generally follow a similar trend, with the cross-dataset performance peaking when the holdout performance peaks at a regularization parameter of $\alpha = 0.00316$. *PIK3CA* is an example of the opposite, a gene where the "smallest good" model tends to outperform the "best" model (Figure 3E). In this case, the peak for the cross-dataset performance occurs at a higher level of regularization (further left on the x-axis), at $\alpha = 0.01$, than the peak for the holdout performance, at $\alpha = 0.0316$. This suggests that a *PIK3CA* mutation status classifier that is more parsimonious, but that has slightly worse performance, does tend to generalize better across datasets to CCLE.

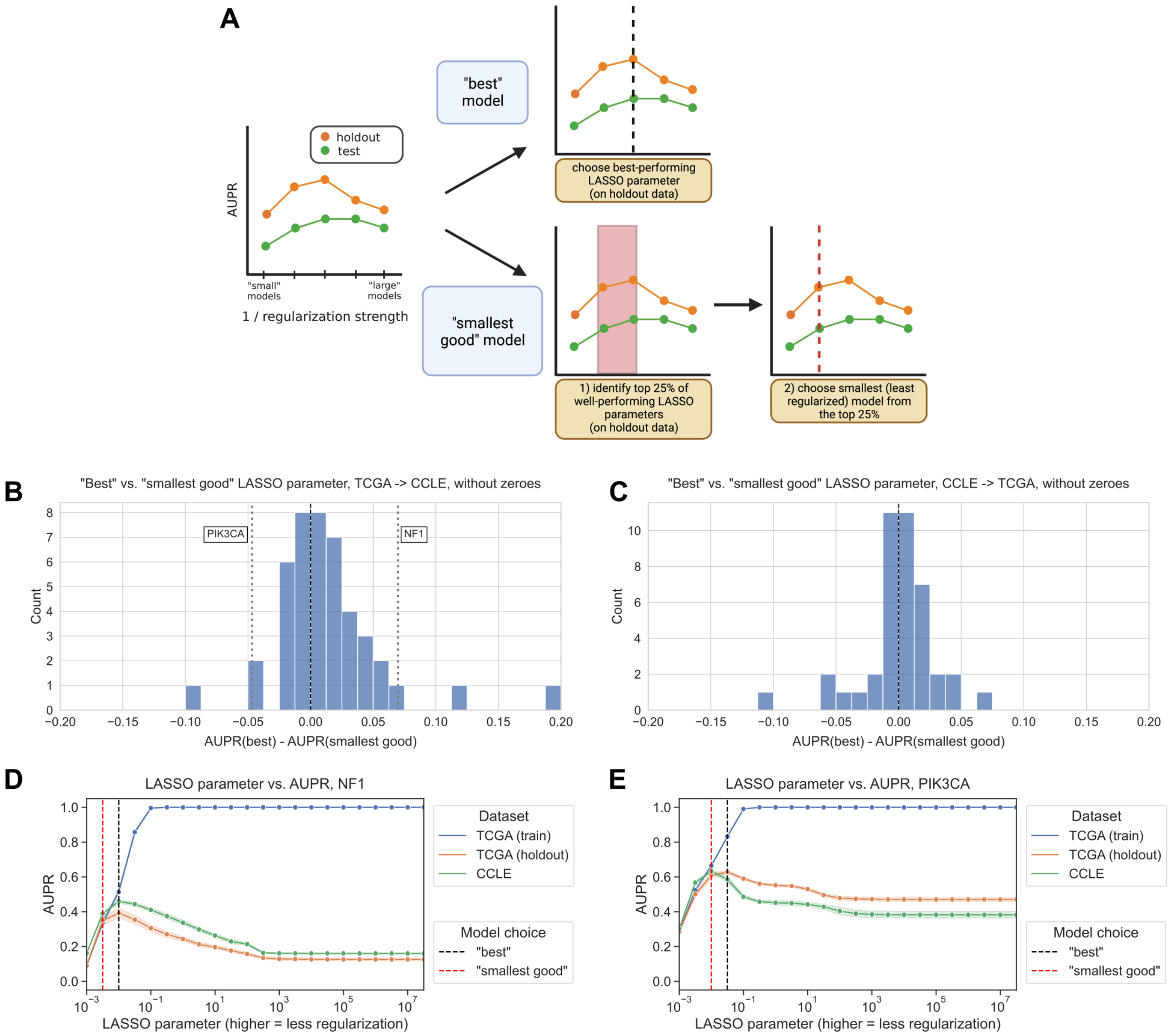

Figure 3: **A.** Schematic of “best” vs. “smallest good” model comparison experiments. **B.** Distribution of performance comparisons between “best” and “smallest good” model selection strategies, for TCGA -> CCLE generalization. Positive x-axis values indicate better performance for the “best” model, negative values indicate better performance for the “smallest good” model. **C.** Distribution of performance comparisons between “best” and “smallest good” model selection strategies, for CCLE -> TCGA generalization. **D.** *NF1* mutation status prediction performance generalizing from TCGA (holdout, orange), to CCLE (green), with “best” and “smallest good” models labeled. **E.** *PIK3CA* mutation status prediction performance generalizing from TCGA (holdout, orange), to CCLE (green), with “best” and “smallest good” models labeled.

References

1. **Scalable Open Science Approach for Mutation Calling of Tumor Exomes Using Multiple Genomic Pipelines**
Kyle Ellrott, Matthew H Bailey, Gordon Saksena, Kyle R Covington, Cyriac Kandoth, Chip Stewart, Julian Hess, Singer Ma, Kami E Chiotti, Michael McLellan, … Armaz Mariamidze
Cell Systems (2018-03) https://doi.org/gf9twn
DOI: 10.1016/j.cels.2018.03.002 · PMID: 29596782 · PMCID: PMC6075717

2. **GISTIC2.0 facilitates sensitive and confident localization of the targets of focal somatic copy-number alteration in human cancers**
Craig H Mermel, Steven E Schumacher, Barbara Hill, Matthew L Meyerson, Rameen Beroukhim, Gad Getz
Genome Biology (2011-04) https://doi.org/dzhjqh
DOI: 10.1186/gb-2011-12-4-r41 · PMID: 21527027 · PMCID: PMC3218867

3. **Evaluating the evaluation of cancer driver genes**
Collin J Tokheim, Nickolas Papadopoulos, Kenneth W Kinzler, Bert Vogelstein, Rachel Karchin
Proceedings of the National Academy of Sciences (2016-11-22) https://doi.org/f9d77w
DOI: 10.1073/pnas.1616440113 · PMID: 27911828 · PMCID: PMC5167163

4. **Regression Shrinkage and Selection Via the Lasso**
Robert Tibshirani
Journal of the Royal Statistical Society: Series B (Methodological) (1996-01)
https://doi.org/gfn45m
DOI: 10.1111/j.2517-6161.1996.tb02080.x

5. **Scikit-learn: Machine Learning in Python**
Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, … Édouard Duchesnay
Journal of Machine Learning Research (2011) http://jmlr.org/papers/v12/pedregosa11a.html

6. **LIBLINEAR: A Library for Large Linear Classification**
Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, Chih-Jen Lin
Journal of Machine Learning Research (2008) http://jmlr.org/papers/v9/fan08a.html

7. **Optimizer's dilemma: optimization strongly influences model selection in transcriptomic prediction**
Jake Crawford, Maria Chikina, Casey S Greene
Cold Spring Harbor Laboratory (2023-06-26) https://doi.org/gsdsvs
DOI: 10.1101/2023.06.26.546586

8. **The effect of non-linear signal in classification problems using gene expression**
Benjamin J Heil, Jake Crawford, Casey S Greene
PLOS Computational Biology (2023-03-27) https://doi.org/gr2q6q
DOI: 10.1371/journal.pcbi.1010984 · PMID: 36972227 · PMCID: PMC10079219

9. **Rectified linear units improve restricted boltzmann machines**
Vinod Nair, Geoffrey E Hinton
Proceedings of the 27th International Conference on International Conference on Machine Learning (2010-06-21) https://dl.acm.org/doi/10.5555/3104322.3104425
ISBN: 9781605589077

10. **PyTorch: An Imperative Style, High-Performance Deep Learning Library**
Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, … Soumith Chintala
arXiv (2019-12-05) https://arxiv.org/abs/1912.01703

11. **Adam: A Method for Stochastic Optimization**
Diederik P Kingma, Jimmy Ba
arXiv (2017-01-31) https://arxiv.org/abs/1412.6980

12. **Cancer Genome Landscapes**
B Vogelstein, N Papadopoulos, VE Velculescu, S Zhou, LA Diaz, KW Kinzler
Science (2013-03-28) https://doi.org/6rg
DOI: 10.1126/science.1235122 · PMID: 23539594 · PMCID: PMC3749880

13. **EGFR Mutations and Lung Cancer**
Gilda da Cunha Santos, Frances A Shepherd, Ming Sound Tsao
Annual Review of Pathology: Mechanisms of Disease (2011-02-28) https://doi.org/dd359s
DOI: 10.1146/annurev-pathol-011110-130206 · PMID: 20887192

14. **Spectrum of EGFR aberrations and potential clinical implications: insights from integrative pan-cancer analysis**
Haijing Liu, Bo Zhang, Zhifu Sun
Cancer Communications (2020-01) https://doi.org/ghsz4b
DOI: 10.1002/cac2.12005 · PMID: 32067422 · PMCID: PMC7163653

15. **Patient-derived cells from recurrent tumors that model the evolution of IDH-mutant glioma**
Lindsey E Jones, Stephanie Hilz, Matthew R Grimmer, Tali Mazor, Chloé Najac, Joydeep Mukherjee, Andrew McKinney, Tracy Chow, Russell O Pieper, Sabrina M Ronen, … Joseph F Costello
Neuro-Oncology Advances (2020-01-01) https://doi.org/gsfw2p
DOI: 10.1093/noajnl/vdaa088 · PMID: 32904945 · PMCID: PMC7462278